BAB II

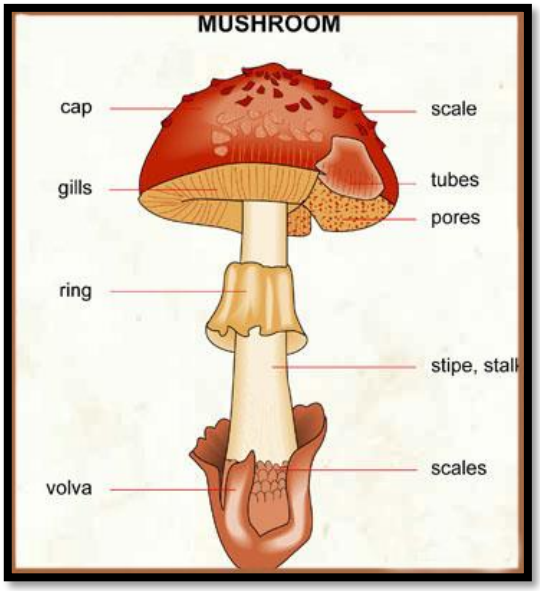
LANDASAN TEORI

2.1. Jamur

Menurut Wiardani (2010), jamur adalah salah satu jenis tumbuhan yang tidak mandiri karena kehidupannya selalu tergantung pada organisme lain sehingga disebut heterotrofik. Karena tidak memiliki klorofil, jamur tidak mampu menghasilkan makan sendiri. Jamur mengambil zat-zat makanan seperti selulosa, glukosa, lignin, protein, dan senyawa pati dari organisme lain. Dengan bantuan enzim yang diproduksi oleh hifa (bagian jamur yang bentuknya seperti benang halus, panjang dan kadang bercabang), bahan makanan tersebut diurai menjadi senyawa yang dapat diserap untuk pertumbuhan.

Nathan Wilson (2013) menegaskan bahwa jamur memiliki kemiripan peran dengan bunga atau buah dalam tanaman. Beberapa bagian dari jamur menghasilkan spora yang mirip dengan serbuk sari atau biji. Dari kemiripan ini juga dapat ditafsirkan bahwa jamur memiliki sifat yang mirip, yakni terdapat jamur yang beracun dan jamur yang tidak beracun atau layak makan. Dalam melakukan identifikasi, Abulude (2013) menjelaskan bahwa lebih baik menggunakan referensi karakteristik yang dimiliki setiap jamur. Karakteristik ini terdiri dari:

a. Ukuran, warna, konsistensi topi dan tangkai.
b. Model dari pelengkap pada insang (yang berada dibalik topi) sampai ke tangkai.
c. Warna spora dalam jumlah yang banyak.
d. Tes reaksi dan kimia.



Gambar 2.1. Struktur Karakteristik Jamur
(Carluccio, 2013)

2.2. Data Mining

Menurut Witten dan Frank (2011), *data Mining* adalah proses ektraksi kumpulan data-data untuk mendapatkan informasi penting yang sebelumnya tidak diketahui. *Data Mining* dilakukan untuk mencari pola di dalam data yang nantinya dapat menghasilkan prediksi yang akurat pada data di masa yang akan datang.

Larose (2005) membagi *data mining* menjadi 6 kelompok berdasarkan tugas yang dapat dilakukan, yaitu:

a. Description

Deskripsi biasa akan dilakukan oleh peneliti atau analis yang secara langsung mencoba mencari cara untuk menggambarkan pola dan kecenderungan yang terdapat dalam data. Kualitas bagus dari penggambaran pola dan kecenderungan dapat diselesaikan dengan *exploratory data analysis*, metode grafis yang mengeksplorasi data untuk mencari pola dan kecenderungan data.

b. Estimation

Estimasi hampir sama dengan klasifikasi, kecuali variabel target estimasi lebih kearah numerik daripada kearah kategori. Model dibangun menggunakan *record* lengkap yang menyediakan nilai dari variabel target nilai prediksi. Selanjutnya, pada peninjauan baru, estimasi nilai dari variabel target dibuat, berdasarkan nilai dari variabel yang memprediksi hasilnya.

c. *Classification*

Di dalam klasifikasi, terdapat variabel sebagai kategori dari target. Seperti kelas yang dapat dibagi menjadi dua atau lebih subset. Model *data mining* akan memeriksa data dalam jumlah besar seperti *dataset* dan *records*, masing – masing data tersebut berisi informasi tentang variabel sebagai kategori dari target serta set input atau variabel prediktor.

d. *Prediction*

Prediksi mempunyai kemiripan dengan klasifikasi dan estimasi, namun hasil nilai dari prediksi hanya akan terlihat nanti di masa mendatang. Dengan kemiripan tersebut, beberapa metode dan teknik yang digunakan dalam klasifikasi dan estimasi dapat dilakukan juga untuk prediksi, hanya dengan keadaan yang tepat untuk melakukan prediksi.

e. *Clustering*

Pengklusteran cukup mengacu kepada pengelompokkan data *records*, pengamatan, atau kasus menjadi kelas – kelas objek yang memiliki kemiripan satu sama lain. Dapat disimpulkan bahwa, kluster merupakan kumpulan data tersebut yang memiliki kemiripan satu sama lain namun tidak mirip atau memiliki perbedaan dengan data dalam kluster lain. Pengklusteran tidak mencoba untuk melakukan klasifikasi, estimasi, ataupun prediksi nilai dari variabel kelas target. Algoritma pengklusteran hanya melakukan pembagian terhadap seluruh data menjadi kelompok – kelompok yang memiliki kemiripan. Kemiripan data dalam suatu kelompok akan bernilai maksimal, sedangkan kemiripan dengan data dalam kelompok lain akan bernilai minimal.

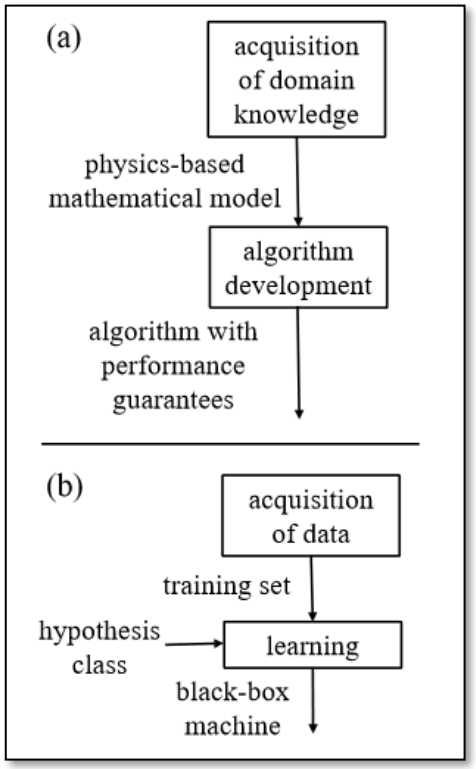
f. *Association*

Tugas dari asosiasi untuk *data mining* adalah mencari atribut yang dapat berdampingan agar dapat mengungkap *rules* untuk mengukur hubungan antara dua atau lebih atribut yang berdampingan.

Dalam bukunya, Maimon, dkk. (2010) menyebutkan bahwa *data mining* merupakan inti dari *Knowledge Discovery in Database* (KDD), KDD adalah proses terorganisir untuk mengidentifikasi pola yang valid, baru, berguna, dan mudah dimengerti dari kumpulan data yang besar dan kompleks.

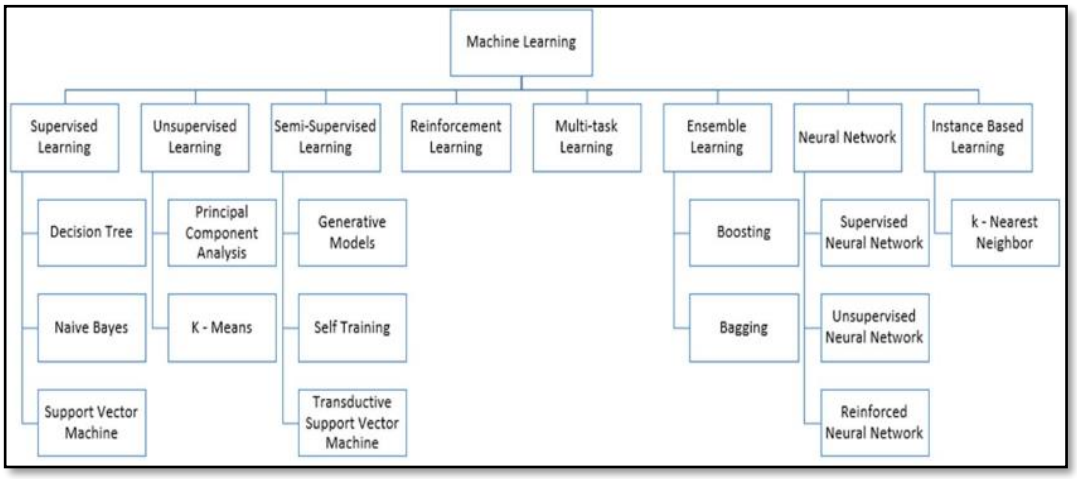
2.3. *Machine Learning*

Machine learning atau pembelajaran mesin memungkinkan pengguna untuk memberikan ilmu kepada komputer agar komputer dapat menganalisa dan membuat rekomendasi ataupun keputusan berdasarkan data yang diberikan sehingga merupakan pilihan tepat untuk melakukan klasifikasi (Cropper, 2019). Simeone (2018) menegaskan bahwa untuk menetapkan suatu ide, akan lebih baik jika memperkenalkan metodologi apa yang mirip dengan ruang lingkup pembelajaran mesin sebagai alternatif untuk pendekatan dengan teknik konvensional untuk desain solusi algoritmik seperti yang diilustrasikan pada Gambar 2.2. Menurut Dey (2016), terkadang setelah melihat data, pola masih tak dapat ditafsirkan atau informasi masih sulit untuk diekstrak, sehingga dibutuhkan pembelajaran mesin.



Gambar 2.2 (a) Desain Alur Pendekatan Dalam Teknik Konvensional; dan (b) Baseline Metode Pembelajaran Mesin.
(Simeone, 2018)

Mitchell (1997) menjabarkan bahwa suatu program komputer dikatakan belajar dari pengalaman E (*experience*) yang berhubungan dengan beberapa kelas tugas T (*task*) dan kinerja ukuran P (*performance*), jika kinerjanya pada tugas-tugas di T, yang diukur dengan P, akan meningkat dengan pengalaman E yang dipelajari. Pembelajaran mesin dapat dilakukan dengan beberapa pendekatan yang diilustrasikan Gambar 2.3.

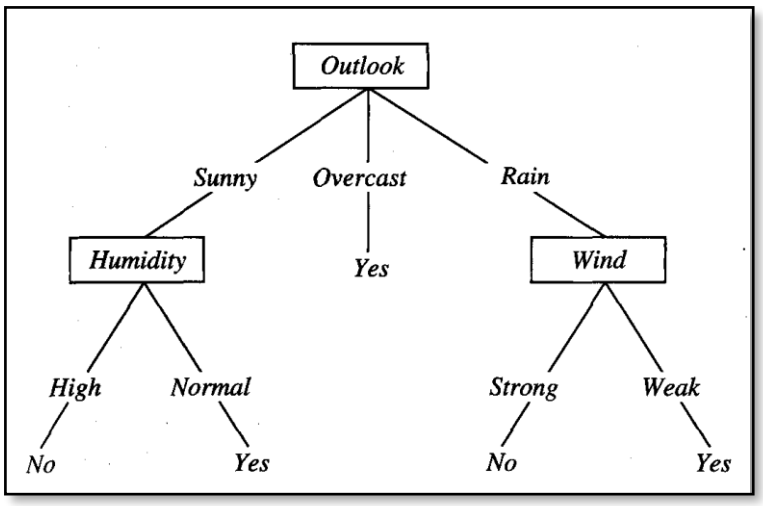


Gambar 2.3 Jenis Pembelajaran Mesin.
(Dey, 2016)

2.4. *Decision Tree*

Decision Tree atau Pohon Keputusan (Mitchell, 1997) merupakan jenis pembelajaran untuk melakukan pendekatan fungsi target bernilai diskrit. Fungsi yang dipelajari akan diwakili oleh pohon keputusan. Pendekatan ini juga dapat direpresentasikan kembali sebagai seperangkat aturan jika-maka (*if-then*) untuk meningkatkan keterbacaan manusia sehingga mudah dipahami pengguna. Dey (2016) menambahkan, bahwa pendekatan ini digunakan terutama untuk tujuan klasifikasi. Setiap pohon terdiri dari daun (*nodes*) dan cabang. Setiap daun mewakili atribut dalam grup yang akan diklasifikasikan dan setiap cabang mewakili nilai yang dapat diambil daun.

Pohon keputusan mengklasifikasikan data dengan menyortirnya dari akar sampai ke beberapa simpul daun, yang menyediakan klasifikasi data. Setiap node dalam tree menentukan tes beberapa atribut data, dan setiap cabang menurun dari daun tersebut berhubungan dengan salah satu nilai yang mungkin untuk atribut ini. Data mulai diklasifikasikan dimulai dari *node* yang berada di akar pohon (*root node)*, melakukan *test* pada atribut data di *node* tersebut dan melanjutkannya ke cabang lainnya. Proses ini diulangi terus hingga *subtree* sudah mencapai akhir pohon (*leaf node*) seperti yang diilutrasikan Gambar 2.4.



Gambar 2.4. Konsep Contoh Pohon Keputusan.
(Mitchell, 1997)

Menurut Alpayadin (2004) pohon keputusan dapat dibagi menjadi tiga bagian, dan dapat dilihat berdasarkan Gambar 2.4.

1. *Root Node*

   Node ini merupakan node yang terletak paling atas dari suatu pohon dan tidak memiliki cabang yang masuk. Contoh : *Node 'Outlook'*.

2. *Internal Node*

   Node ini merupakan node percabangan, hanya terdapat satu input serta mempunyai minimal dua output. Contoh : *Node 'Humidity'*

3. *Leaf Node*

   Node ini merupakan node akhir, hanya memiliki satu input, dan tidak memiliki output. Contoh : *Node 'Yes' 'No'*

Menurut Patil, dkk., (2012) pemilihan atribut dapat dilakukan dengan proses *information gain*. Dalam memilih atribut untuk memecahkan objek harus dipilih atribut yang menghasilkan information gain paling besar. Atribut yang memiliki *information gain* terbesar akan dipilih sebagai *entropy*. Han, dkk. (2012) menjelaskan persamaan yang digunakan dalam mencari *entropy* dan *information gain* seperti Persamaan 2.1, 2.2 dan 2.3.

$$Info(D) = -\sum_{i=1}^{m} P_i log_2(P_i) \quad ...(2\ 1)$$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{D} x Info(D_j) \quad ...(2\ 2)$$

$$Gain(A) = Info(D) - Info_A(D) \quad ...(2\ 3)$$

Keterangan :

- $Info$ = Nilai *entropy*
- D = sampel data
- m = Jumlah kelas yang ada pada atribut
- P_i = Peluang dari kelas i atau rasio dari kelas i.
- $\frac{|D_j|}{D}$ = Bobot partisi ke-j
- $Gain(A)$ = *Information Gain*, yang akan mengukur banyaknya "informasi" yang diberikan suatu fitur tentang *class* atau *label* yang akan dicari

2.5. *Ensemble Learning*

Ensemble Learning (Zhou, 2012) merupakan salah satu pendekatan pembelajaran mesin yang mempunyai beberapa pembelajaran yang akan di-*train* untuk melakukan tugas yang sama. Berbeda dengan pendekatan pembelajaran mesin biasa yang hanya mempelajari satu hipotesa dari data *train*, pendekatan ensemble mencoba membangun beberapa hipotesa dan menggabungkannya untuk digunakan sebagai kesatuan. Pendekatan ini telah diamati dan mendapatkan hasil, yakni rata – rata kombinasi dari beberapa pembelajaran hampir selalu lebih baik dalam melakukan pekerjaan daripada pembelajaran individual. Dey (2016), menyebutkan bahwa terdapat 2 teknik *ensemble learning* yang populer, yakni:

a. *Boosting*

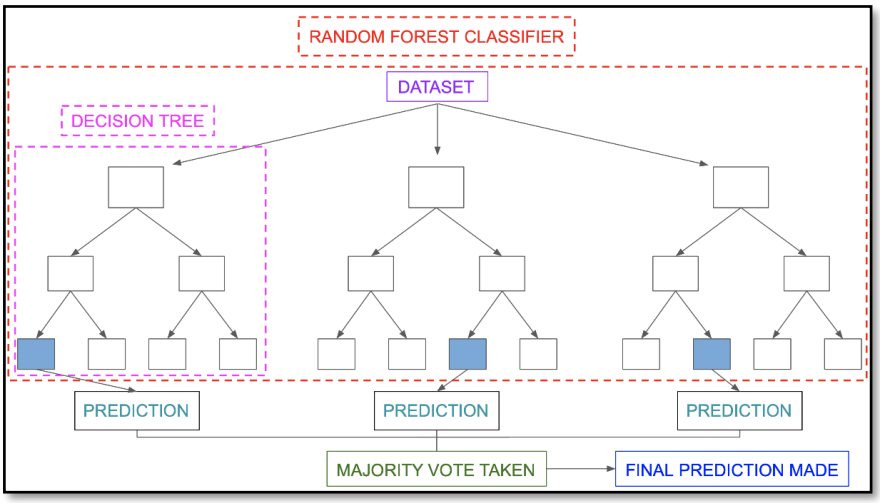
   Boosting adalah teknik dalam *ensemble learning* yang digunakan untuk mengurangi bias dan varians. *Boosting* akan membuat koleksi dari pembelajaran yang lemah menjadi pembelajaran yang kuat. Yang dimaksud dengan pembelajaran lemah adalah pembelajaran yang tidak cocok dan tergolong nyaris tidak berkolerasi dengan klasifikasi yang diperlukan.

b. *Bagging*

   Bagging atau kepanjangan dari *Bootstrap Aggregating* biasa digunakan dimana akurasi dan stabilitas dari pembelajaran mesin perlu ditingkatkan. Ini berlaku untuk pembelajaran berupa klasifikasi dan regresi. *Bagging* juga mengurangi varians dan membantu dalam menangani overfitting.

2.6. *Random Forest*

Menurut Starmer (2018), *Random Forest* (RF) merupakan salah satu *ensemble learning* dari *bagging* sebagai kombinasi *decision tree* yang memberikan akurasi yang tergolong lebih tinggi karena algoritma ini akan menangani *missing values* dan tetap menjaga keakuratan data dengan jumlah besar. Breiman (2001) juga menegaskan di bukunya, bahwa kesalahan pada RF menjadi konvergen berdasarkan jumlah pohon keputusan yang terbentuk dan membuat setiap pohon tergantung pada nilai vektor yang acak diambil secara independen dan dengan distribusi yang sama untuk semua pohon keputusan. Masing–masing *decision tree* akan saling menjaga dari kesalahannya masing-masing dan menyimpulkan hasil klasifikasi sesuai dengan *majority vote* atau prediksi individual *decision tree* seperti ilustrasi Gambar 2.5. Banyaknya *decision tree* yang terbentuk tidak akan membuat algoritma ini menjadi *over-fitting* ataupun menjadi rentan terhadap *error*.



Gambar 2.5. Konsep Contoh *Random Forest*
(Koehersen, 2017)

Pada RF, pembentukan *tree* dapat dilakukan dengan *Gini* dan *Entropy* (Sieling, 2015). *Gini* lebih ditujukan untuk atribut kontinyu sedangkan *entropy* ditujukan untuk atribut diskrit yang terjadi pada kelas. Pembetukan tree merujuk pada persamaan 1, 2, dan 3 sama dengan persamaan *decision tree* (Breiman, 1996). *Decision Tree* dibentuk secara rekursif sampai dengan jumlah yang ditentukan. Menurut Breiman (2001), secara optimal pohon yang dibentuk pada klasifikasi ditentukan dengan $\sqrt{p}$ dan untuk regresi menggunakan *p/3* dimana *p* merupakan jumlah *predictor* yang digunakan. *Output* dari semua pohon dalam melakukan klasifikasi dikumpulkan dalam membuat klasifikasi akhir dilakukan menerapkan persamaan *majority vote* seperti persamaan 2.4 (Hastie, 2008).

$$\hat{C}_{rf}^{B}(x) = majority\ vote\ \{\hat{C}_b(x)\}_1^B \quad ...(2.4)$$

$\hat{C}_b(x)$ merupakan *class* yang akan menjadi target klasifikasi dari *decision tree* sejumlah *b.*

2.7. Evaluasi Performa

Menurut Arthana (2019), dalam mengevaluasi performance algoritma dari *machine learning* (khususnya dengan *supervised learning*), dapat digunakan acuan *Confusion Matrix* untuk memprediksikan kategori. Secara umum, *confusion matrix* dapat dilihat pada Tabel 2.1. Persamaan untuk melakukan perhitungan evaluasi performa dapat dilihat pada Persamaan 2.5, 2.6, 2.7, 2.8, 2.9.

Tabel 2.1 *Confusion Matrix*

Aktual / Prediksi	Positif (1)	Negatif (0)
Positif (1)	*True Positive* (TP)	*False Positive* (FP)
Negatif (0)	*False Negative* (FN)	*True Negative* (TN)

(Prasetyo, 2014)

Berdasarkan tabel di atas, kelompok hasil prediksi secara umum diartikan sebagai:

- *True Positive* (TP) = teridentifikasi secara benar
- *False Positive* (FP) = teridentifikasi secara salah
- *True Negative* (TN) = tertolak secara benar
- *False Negative* (FN) = tertolak secara salah

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad ...(2.5)$$

$$Precission = \frac{TP}{TP+FP} \quad ...(2.6)$$

$$Recall = \frac{TP}{TP+FN} \quad ...(2.7)$$

$$Specificity = \frac{TN}{TN+FP} \quad ...(2.8)$$

$$F1\ Score = \frac{2*(Recall*Precission)}{Recall+Precission} \quad ...(2.9)$$

Accuracy merupakan ketepatan prediksi (bernilai *true*). *Precission* merupakan kemampuan pengujian untuk mengidentifikasi kelas positif yang di prediksi dengan tepat yang benar. *Recall* merupakan kemampuan pengujian untuk mengidentifikasi nilai positif dari sejumlah data diprediksi dengan tepat. *Specificity* merupakan kemampuan pengujian untuk mengidentifikasi hasil negatif dari sejumlah data yang memang negatif. *F1 Score* merupakan nilai untuk membantu mengukur *recall* dan *precission* bersamaan.